Design, Construction, and Testing of an Electrodynamic Uniaxial Bench-Scale Shake Table

Bill Wu

Advisors: Professor Erik Cheever and Professor Faruq Siddiqui

Department of Engineering
Swarthmore College
May 8th, 2015

Abstract
A bench-scale shake table was designed and constructed for the Swarthmore College Engineering Department. The frame of the shake table was made of Unistrut. It's powered by a motor with a complete built-in servo control system that takes analog output as the velocity through an A/D converter. Two transfer functions were developed: one that connects the motor to the base plate and the other connecting the plate to a simple test structure (lumped mass model). Sensors (LVDT and accelerometers) were hooked onto the base plate and structure to measure acceleration and displacement data to be sent back to Matlab for further analysis and plotting. We used Matlab's linear simulator (lsim) to calculate the theoretical output for a known input waveform, and compared the results to the actual measured output (by the accelerometer). The results matched well for impulse functions, step functions, and arbitrary waveforms such as the El Centro earthquake. Two multi-story stick mass systems were shaken at their respective resonant frequencies observe the effects of resonance.

Acknowledgements

I would like to thank Professor Cheever for his patience and wisdom along every single step of this project. I also want to thank Professor Siddiqui for his guidance on implementing testing structures, Professor Everbach for helping me understand the previous codes developed for the motor at the start of the semester, Grant Smith and James Johnson for their patience, guidance, and support in the machine shop, a place I now no longer dread.

Table of Contents

Abstract
Acknowledgement
Table of Contents
1. Introduction
    1.1Objective and Motivation
    1.2 Key Components of the Shake Table
2. Construction
3. Theory
    3.1 Derivation: Transfer function of a Second Order Underdamped System in time and Laplace domain
    3.2 Theoretical calculation: finding the resonant frequency of a lumped mass system
    3.3 Procedures for frequency domain analysis
    3.4 Theoretical calculation: finding the resonant frequencies of a two-story stick mass building
4. Animatics SmartMotor
    4.1 Communication
    4.2 Programming Notes
    4.3 Motion Commands
    4.4 I/O Ports
    4.5 Analog Velocity
5. Data Acquisition (DAQ)
6. Calibration
7. Testing Structures
    7.1 Lumped Mass System
    7.2 Two-story Stick Mass Model
8. Results
    8.1 The Big Picture
    8.2 Comparing command input with measure velocity output
    8.3 Obtaining *H*plate
    8. 4 Obtaining *H*structure
    8.5 Discussion on the significance of *H*plate
    8. 6 Using Matlab's LSIM
    8. 7 Addition of a dashpot to enhance the El Centro simulation results
    8.8 FFT analysis of two multistoried stick mass structure
9. References
Appendices
    Appendix A: Matlab script file—Analog Velocity
    Appendix B: Matlab script file—FFT Procedure

List of Figures

Figure 1: Animatics SmartMotor SM23165DT
Figure 2: National Instrument PCI-6221
Figure 3: A/D Convertor Box
Figure 4: Linear Variable Differential Transformer (LVDT)
Figure 5: PCB 352B70 Accelerometer
Figure 6: Unistrut™ design and dimension of the shake table
Figure 7: LVDT stand
Figure 8: Dashpot stand
Figure 9: Mounting the SmartMotor onto Unistrut™
Figure 10: Panoramic view of the shake table upon completion
Figure 11: Idealized Single DOF lumped mass structure used for testing
Figure 12: FFT of El Centro acceleration response
Figure 13: Multimass-spring model for a two-story shear building
Figure 14: Frequency domain response to an impulse of a 2-story structure
Figure 15: RS-232 Through USB for D-Style Motors MOOG Animatics (2014). SmartMotor User's Guide
Figure 16: SmartMotor Interface example screenshot
Figure 17: Linear and sinusoidal motion
Figure 18: SmartMotor Connector Pinouts
Figure 19: Setup of the DAQ system
Figure 20: Test Run for calibration
Figure 21: Lumped mass structure used for testing
Figure 22: Two-story Stick Mass System
Figure 23: Connections details of the two-story stick mass system
Figure 24: The Big Picture
Figure 25: Command input and recorded input for an impulse
Figure 26: Command input and recorded input for a sinusoid
Figure 27: Command input and recorded input for the El Centro
Figure 28: Zoom-in of Figure 27
Figure 29: cftool to obtain curve fit for Hplate
Figure 30: Response to Impulse
Figure 31: Curve fit for Hstructure
Figure 32: Content of El Centro captured by Hplate
Figure 33: Theoretical and measured output for an impulse
Figure 34: Theoretical and measured output for a sinusoid
Figure 35: Theoretical and measured output for El Centro
Figure 36: Response to an impulse for more heavily damped system
Figure 37: Theoretical and measured output for El Centro for heavily damped system
Figure 38: Time and frequency domain responses to an impulse of a 2-story structure
Figure 39: Time and frequency domain responses to an impulse of a 3-story structure

1. Introduction

1.1 Objectives and motivation

A bench-scale shake table can be a meaningful to study a model's response to a variety of dynamic loading, including earthquakes, which are arbitrary waveforms. The study of earthquakes is important because earthquakes cause billions of dollars in damage every year around the world and tens of thousands of deaths and injuries. In the United States alone, around 1,200 deaths have been recorded since 1900. Many more fatalities occurred in earthquakes elsewhere.

Buildings are normally adequately designed for gravity and vertical loads. Thus, lateral movements provided by the uniaxial shake table can introduce bending to the columns, and torsion if the center of mass and center of resistance are offset. Testing of models of actual buildings and building prototypes is one method that is useful in understanding the forces at work. Models built of plywood and aluminum all-thread allow us to see the seismic behavior of a structure, and to understand how the period of an earthquake if it is resonant with a building period, will cause the most damage or even collapse.

The shake table is a device that simulates a dynamic loading or a seismic event. It can also be used to create fictional "worst case" scenarios or resonant frequencies. In computer controlled shake tables a computer program generates a signal, and a digital signal is sent to a digital/analog converter, which sends a voltage to the amplifier. The amplifier amplifies the voltage and sends it to the shaker platform to which the model is attached. The constructed shake table is a one-degree of motion shake table, meaning that it will move only in one lateral direction. A model on a shake table with the same stiffness or resonant frequency as the prototype building, will act in a way similar to that of the actual building.

1.2 Key components of the shake table

Motor: Animatics SmartMotor SM23165DT

The SM23165DT motor with HLD60 Internal Rollers and a stroke length of 400 mm was chosen. The carriage shown in Figure 1 moves through a belt driven system classified as a harmonic linear drive.

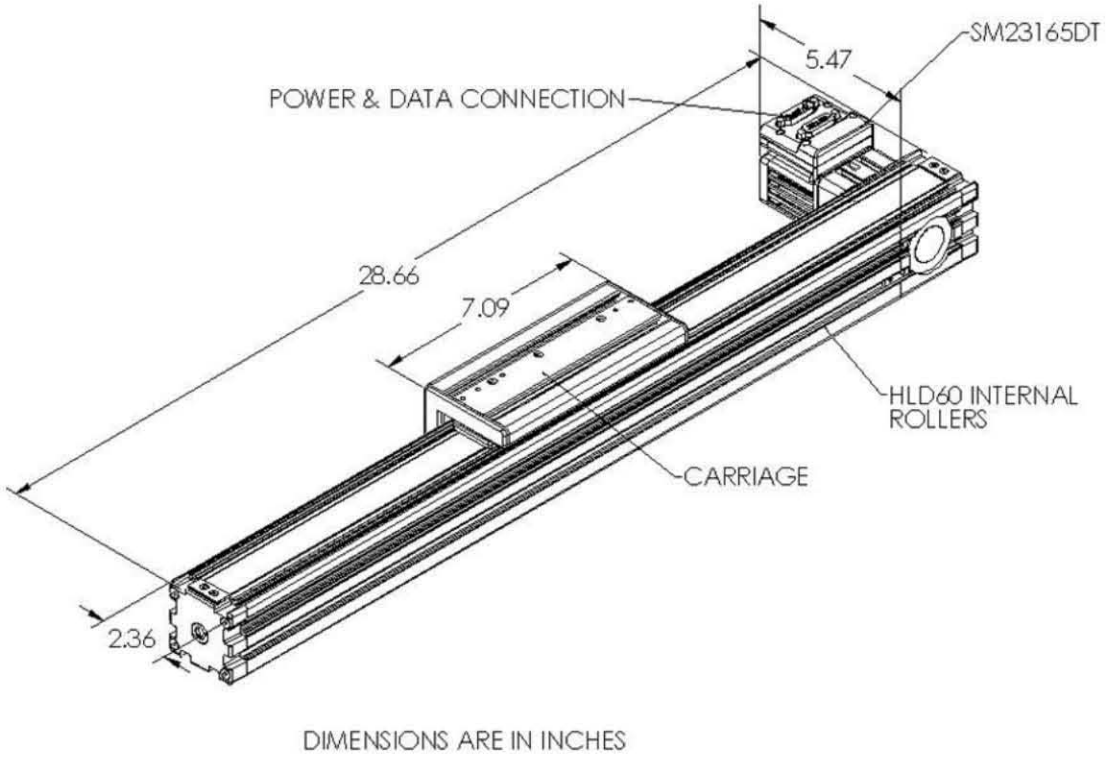
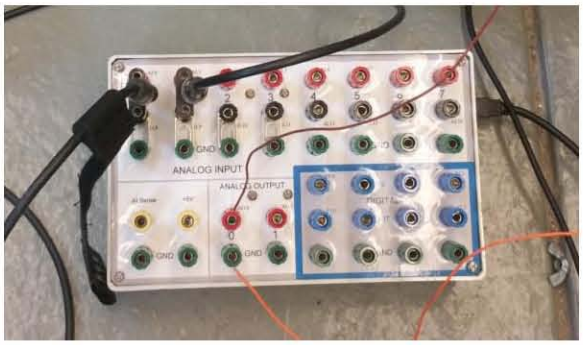


Figure 1: Animatics SmartMotor SM23165DT

Data Acquisition (DAQ) system:

For our DAQ, we used a National Instrument PCI-6221, shown below:



Figures 2 and 3: NI PCI-6221 and its placement inside an A/D convertor

We like it for its Correlated DIO, which enables digital and analog functions to be synchronized with hardware-timed precision and reduced the need to write an internal timer inside Matlab.

LVDT—Linear Variable Differential Transformer

The following figure depicts our LVDT:

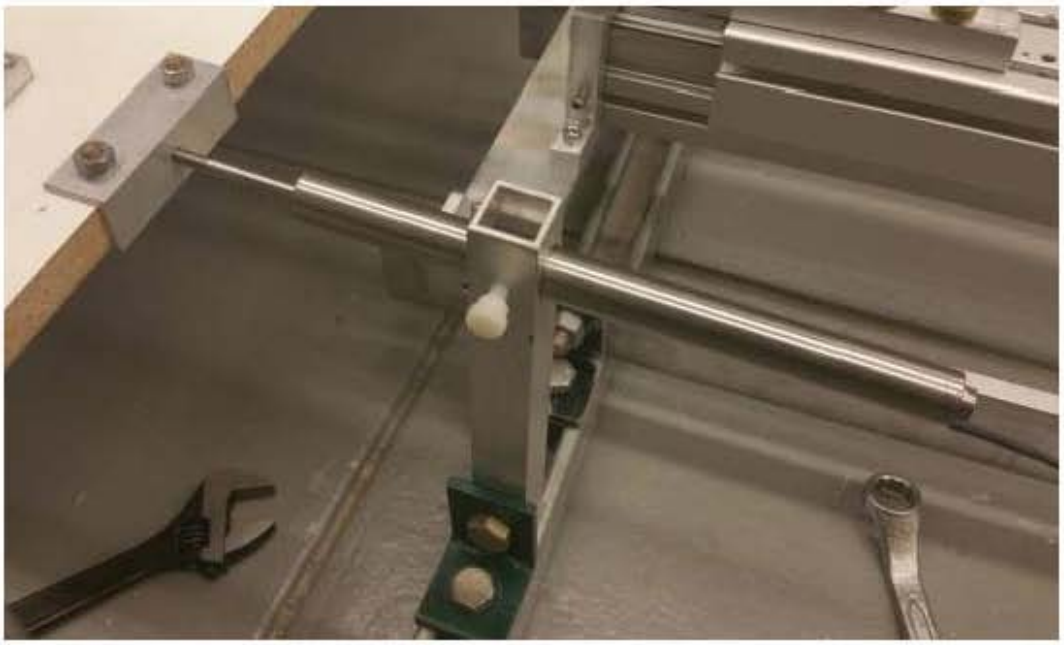


Figure 4: Linear Variable Differential Transformer (LVDT)

Linear variable differential transformer (LVDT) is a type of electrical transformer used for measuring linear displacement. LVDT's are inherently frictionless, and converts a position or linear displacement from a mechanical reference into a proportional electrical signal containing phase and amplitude through induction current. The calibration ratio is $1V = 10mm$.

Accelerometer

The following figure depicts out accelerometer:

Figure 5: PCB 352B70 Accelerometer

The accelerometer is a device that measures proper acceleration ("g-force"). The calibration ratio for our specific model is $1V = \frac{98.1m}{s^2}$.

2. Construction

Construction of the shake table consisted of several key components, and a few pieces of add-ons to support future additions to the shake table. The main frame was first erected using Unistrut™ pieces. Then, four pieces of cable hung from the top frame pieces to support the base plate, which is a piece of plywood. The base plate is then connected to the motor. Holders for the LVDT and dashpot were later constructed and stabilized onto Unistrut™ pieces.

A Unistrut™ metal framing system was deemed optimal for the shake table's frame as it can be used to support a large load while also be taken apart easily for storage. The Unistrut™ system contains specially configured "lipped channels" made of carbon steel that can be connected in a variety of configurations using fabricated Unistrut™ fittings.

Solidworks, a 3D CAD modeling software was used to design the flume. The various Unistrut™ channels and fittings were obtained online from the Unistrut™ CAD Library. Different configurations of the channels were then considered to be built around the dimensions of the flume. The final design of the frame is shown in Figure
.

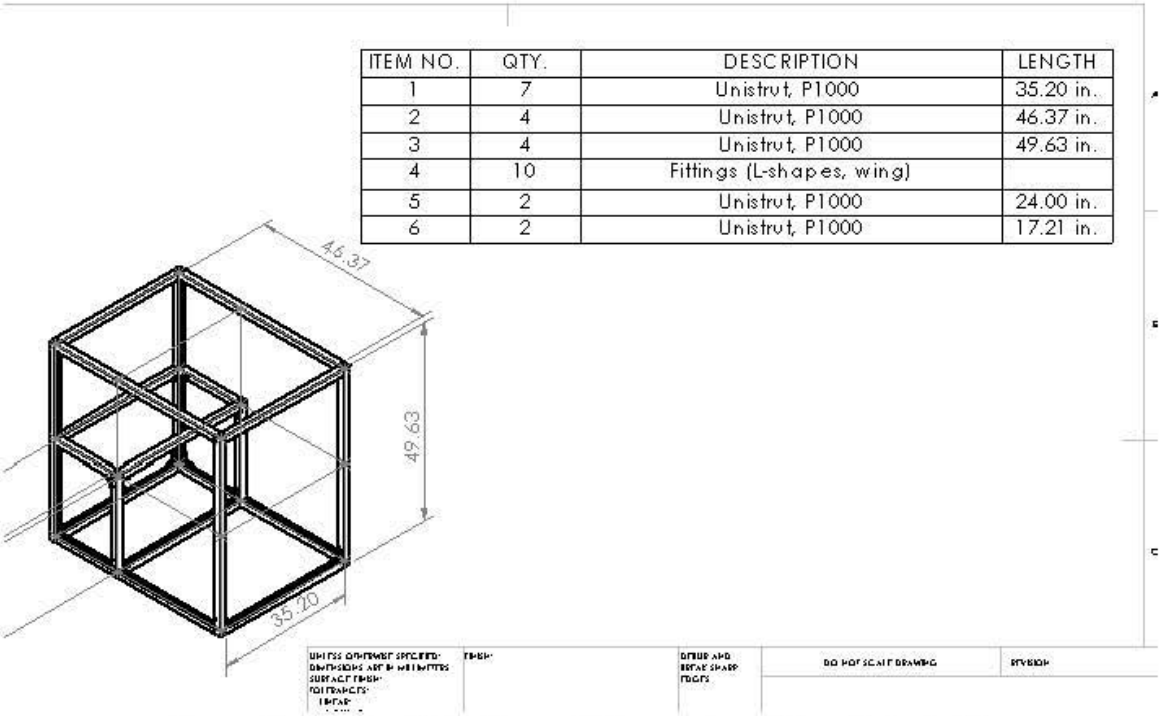

ITEM NO. | QTY. | DESCRIPTION | LENGTH
1 | 7 | Unistrut, P1000 | 35.20 in.
2 | 4 | Unistrut, P1000 | 46.37 in.
3 | 4 | Unistrut, P1000 | 49.63 in.
4 | 10 | Fittings (L-shapes, wing) |
5 | 2 | Unistrut, P1000 | 24.00 in.
6 | 2 | Unistrut, P1000 | 17.21 in.
46.37
49.63
35.20
DO NOT SCALE DRAWING

Figure 6: Unistrut™ design and dimension of the shake table

The holder for the LVDT is made of a hollow section aluminum bar, as shown in Figure 7 :

Figure 7: LVDT stand

The easiness to make additions on Unistrut™ is manifested. The rod of the LVDT is fastened to the base plate through a tapped hole. Also note that once the rod of the LVDT is fastened to the base plate, there is a physical limitation to displacement of the base plate. This won't cause too much inconvenience since our position waveforms (integrated from velocity) is near symmetrical about zero so displacement of the base plate is negligible.

The holder for the dashpot is shown below:

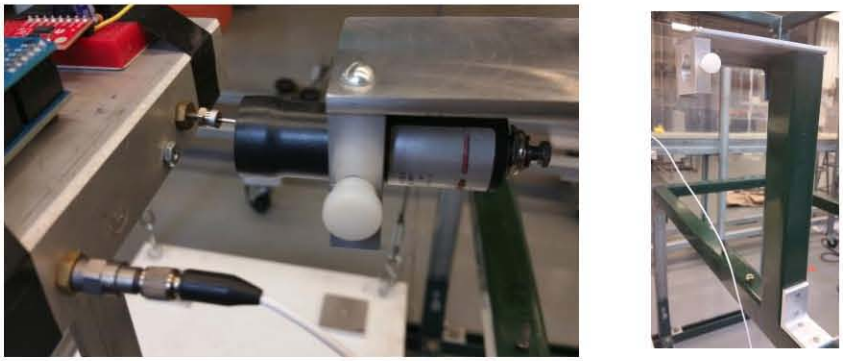


Figure 8: Dashpot stand

The setscrews on the SmartMotor were carefully removed and preserved as we detached the motor from the wave flume. The SmartMotor was mounted onto the frame through two vertical aluminum beams, as shown in Fig. :

Figure 9: Mounting the SmartMotor onto Unistrut™

The frame, cables, SmartMotor, base plate, and LVDT stand were completed within the first three to four weeks of the semester. The need for a dashpot holder came around the 12th week, by which time it was built by "J" and attached onto the structure. Upon completion, the complete view of the shake table is shown in Figure 10:

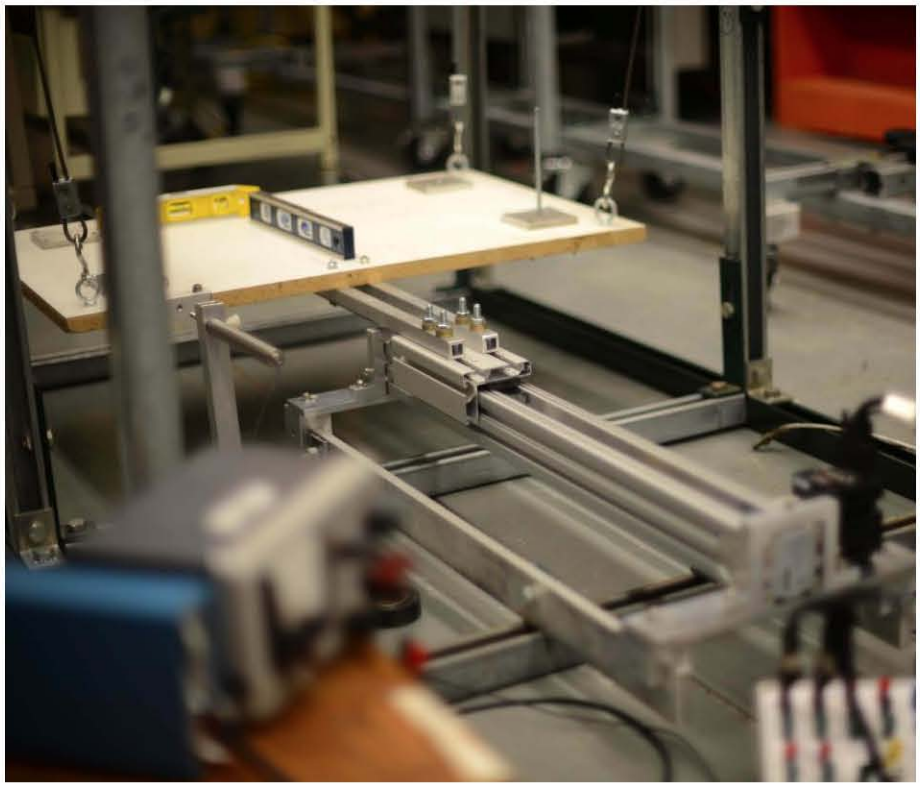


Figure 10: Panoramic view of the shake table upon completion

3. Theory

3.1 Derivation: Transfer function of a Second Order Underdamped System in time and Laplace domain

Consider a mass-spring-damper system, with an input of $x(t)$. Equation of motion will establish the following:

$$m\frac{d^2y}{dt^2} + c\frac{dy}{dt} + ky = x(t).$$

Define the damping variable to be c:

$$c = 2\zeta w_0 m.$$

Also recall that the relationship for resonant frequency is $w_0 = \sqrt{\frac{k}{m}}$. Substituting the above two equations into the equation of motion:

$$\frac{d^2y}{dt^2} + 2\zeta w_0 \frac{dy}{dt} + w_0^2 y = K w_0^2\, x(t).$$

Performing Laplace transform on both sides of the equation we get:

$$Y(s)[s^2 + 2\zeta w_0 s + w_0^2] = K w_0^2 X(s)$$

Rearrange to get:

$$H(s) = \frac{X(s)}{Y(s)} = \frac{K w_0^2}{s^2 + 2\zeta w_0 s + w_0^2},$$

which is the transfer function for a second-order underdamped system when the input and output are both in terms of velocity.

Similarly, the transfer function for velocity in, acceleration out can be derived:

$$H(s) = \frac{K w_0^2 s}{s^2 + 2\zeta w_0 s + w_0^2}.$$

To obtain these transfer functions in the time domain, we can set up all the variables using Matlab's symbolic toolbox and perform inverse Laplace through Matlab's '`iLaplace`' command.

The time domain transfer function for velocity in velocity out is:

$$H(t) = 1 - \frac{1}{\sqrt{1-\zeta^2}} e^{-\zeta\omega t} \sin\left(\omega\sqrt{1-\zeta^2}\, t + \operatorname{acos}(\zeta)\right).$$

The time domain transfer function for velocity in acceleration out is:

$$H(t) = w_0^2 e^{-\zeta\omega t} \cosh\left(\omega t\sqrt{1-\zeta^2}\right) - \frac{\zeta \sinh(\omega t\sqrt{1-\zeta^2})}{\sqrt{1-\zeta^2}}.$$

3.2 Theoretical calculation: finding the resonant frequency of a lumped mass system

The lumped mass system used and the cross section of the cantilever beam is shown in Figure :

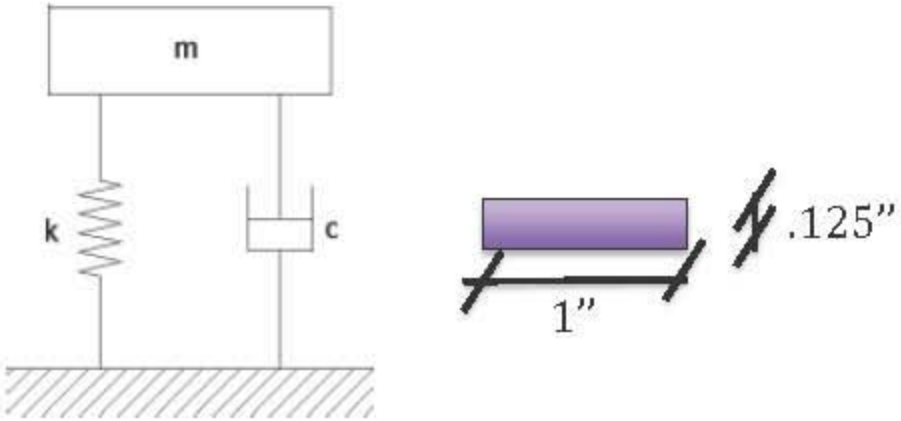


Figure 11: Idealized Single DOF lumped mass structure used for testing

The second moment of area is:

$$I = \frac{1}{12} bh^3 = \frac{1}{6144}\, in^4.$$

The elastic modulus of aluminum is $E = 10(10^6)$psi, the weight of the aluminum block (with the attached SparkFun accelerometer and Arduino) is $W = 1.905$ lbs, the length of the cantilever beam is $L = 26''$. Thus, we find the spring constant of the lumped mass system:

$$k = \frac{3EI}{L^3} = .278 \frac{\text{lbs}}{\text{in}}.$$

We can then find the resonant frequency ω_0:

$$\omega_0 = \sqrt{\frac{kg}{W}} = 1.19 \text{ Hz.}$$

3.3 Procedures for frequency domain analysis

(Sample code for this procedure is attached in Appendix B)

```
FFT = fftshift(fft(baseAcce));
dt=time(2)-time(1);
f=((1:length(time))-(baseAcce/2))/dt/baseAcce;
plot(f,abs(FFT))
```

Performing this analysis on the El Centro acceleration response, we obtain the following frequency-amplitude correspondence:

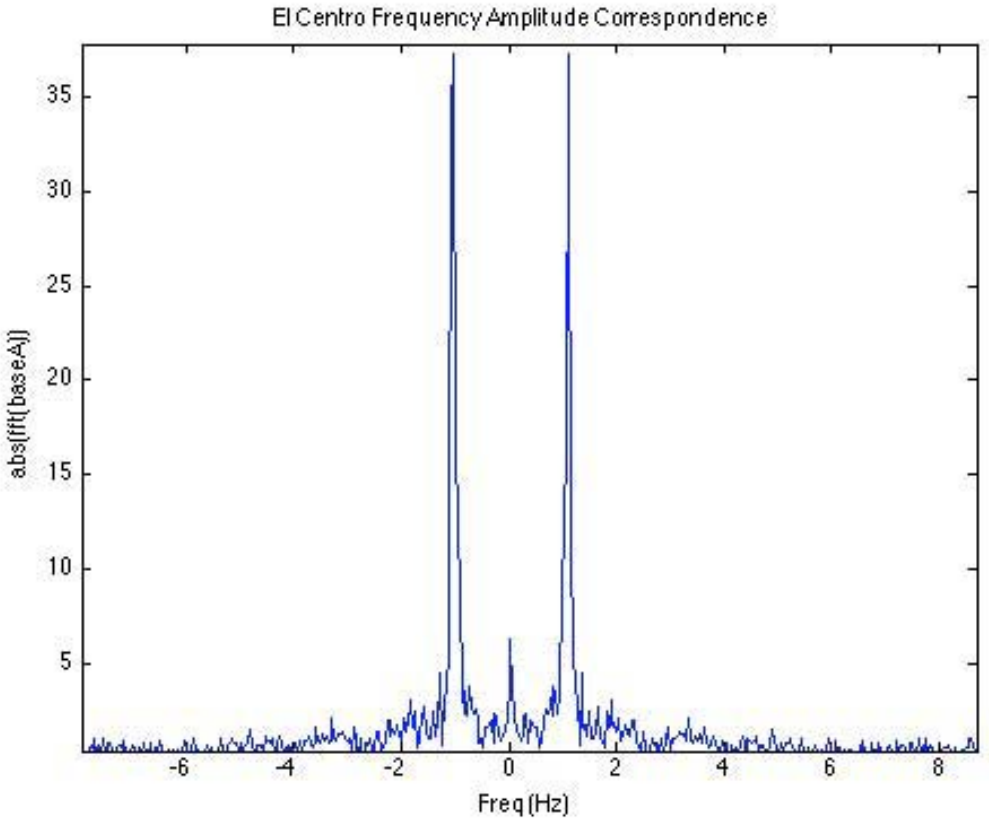


Figure 12: FFT of El Centro acceleration response

3.4 Theoretical calculation: finding the resonant frequencies of a two-story stick mass building

The building to be analyzed is the simple steel rigid frame shown in Fig. . The weights of the floors are the same and measured out before assembled. It is further assumed that the structural properties are uniform along the length of the building.

We model the building as a two-story shear building, which can be represented by the spring-mass system shown in Fig. .

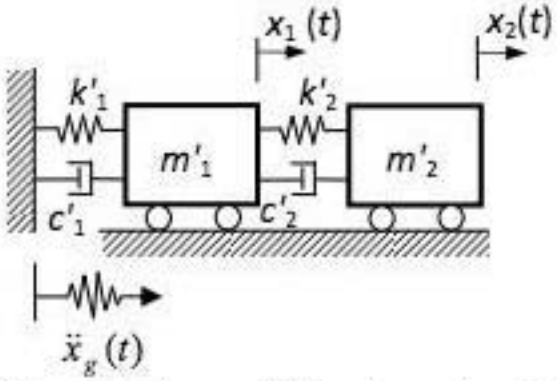


Figure 13: Multimass-spring model for a two-story shear building

The weight is weighed in lbm:

$$W_1 = W_2 = 1\text{lb}$$

The mass can be calculated:

$$m_1 = m_2 = .0026\text{lb}\,s^2/\text{in}$$

Since the girders are assumed to be rigid, the stiffness (spring constant) of each story is given by:

$$k = \frac{12E(4I)}{L^3} = 16.3\ \text{lb/in},$$

and the individual values for the all-threads indicated are thus:

$$k_1 = k_2 = k = 16.3\frac{\text{lb}}{\text{in}}$$

The equations of motion for the system are obtained:

$$m\ddot{y}_1 + k_1 y_1 - k_2(y_2 - y_1) = 0,$$
$$m\ddot{y}_2 + k_2(y_2 - y_1) = 0.$$

In the usual manner, these equations of motion are solved for free vibration by substituting:

$$y_1 = a_1 \sin(\omega t - \alpha)$$
$$y_2 = a_2 \sin(\omega t - \alpha)$$
$$\ddot{y}_1 = -a_1\omega^2 \sin(\omega t - \alpha)$$
$$\ddot{y}_2 = -a_2\omega^2 \sin(\omega t - \alpha)$$

The expansion of the determinant of the matrix formation of the equations of motion gives a quadratic equation in ω^2, namely,

$$m^2\omega^4 - 3mk\omega^2 + k^2 = 0.$$

Substitution of the numeric values just obtained:

$$m^2\omega^4 - 3mk\omega^2 + k^2 = 0.$$

Therefore, the natural frequencies of the structure are:

$$w_1 = 8.69\text{Hz};\ w_2 = 20.59\ \text{Hz}.$$

Recall the FFT analysis of the data obtained via an impulse on the two-story building:

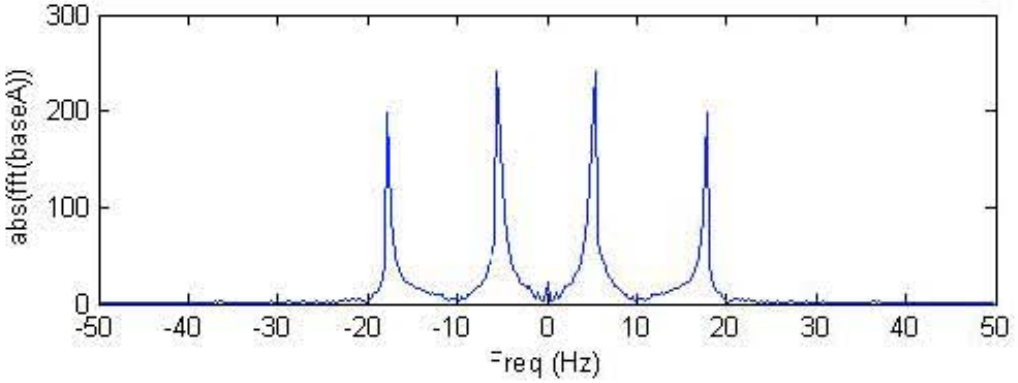


Figure 14: Frequency domain response to an impulse of a 2-story structure

The actual natural frequencies are:

$$w_1 = 6.1\text{Hz};\ w_2 = 19.2\text{Hz}$$

4. Animatics SmartMotor

4.1 Communication

The Animatics SmartMotor is electrically driven which enables the creation of arbitrary waveforms as opposed to only hardcoding waveforms using discrete data points of velocity, position, etc. The motor communicates through a PC serial port via a RS232 port. A single connection to the 7-Pin Combo D-Sub Power & I/O provides power and communication to the motor using a CBLSM1-xM cable. The motor can be powered using a 24-48 VDC power supply. An Agilent N5746A power supply was used which has a DC output voltage rating of 40 V, 19 A current and 760 W power. A schematic of the communication setup between a PC and the motor is shown in Figure 15:

Figure 15: RS-232 Through USB for D-Style Motors MOOG Animatics (2014). SmartMotor User's Guide

4.2 Programming Notes

Due to the direct connection of the motor to a serial port, the motor can be controlled using the MOOG Animatics SmartMotor interface (SMI) Software or Matlab. SMI enables easy communication through the terminal program which provides immediate response to a given command. Additionally the SMI software contains debugging options and can obtain information from the motor including current position, velocity, acceleration and voltage. Because of these characteristics, the SMI software is ideal for troubleshooting. In addition to the SMI software, Matlab can be used to communicate with the motor by creating a serial port and then sending text files through the serial port to the SmartMotor using the 'fprintf' command. A screenshot of the SMI programming interface is shown in Figure 16:

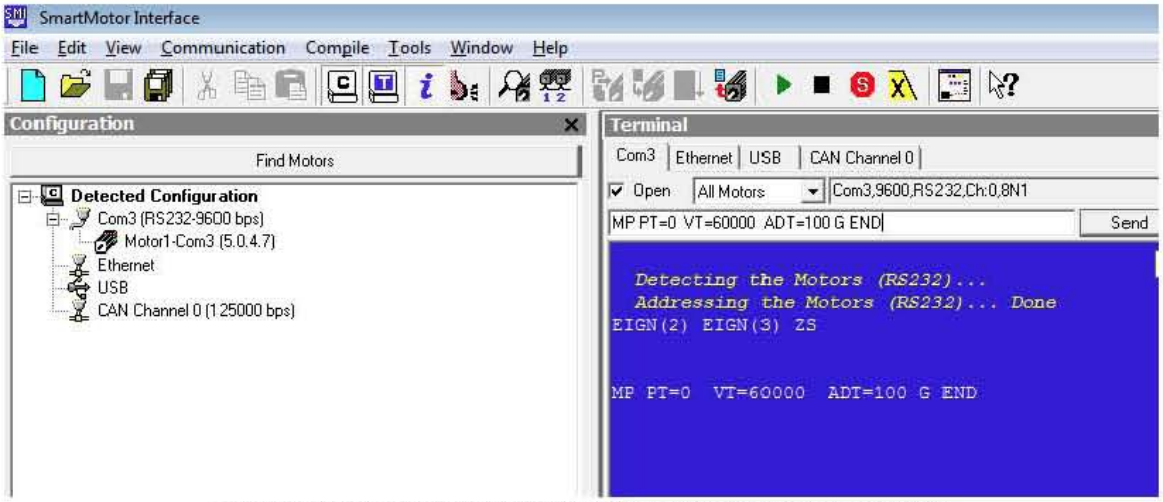


Figure 16: SmartMotor Interface example screenshot

In configuration, the connection between the motor to the PC com-port is established through 'Com3'. The terminal window acts like Matlab's command window. Commands can be given to control the motor.

allows us to upload a script file onto SmartMotor's hard-drive. Only one program can be stored at a time. With each upload, the previous version is erased and replaced.

Prior to moving the motor, the over-travel limits and fault bits must be cleared. To clear the over-travel limits, the EIGN() command , "Enable Inputs as General Use", is used. The historical fault bits are cleared by using the ZS command. The following command lines must therefore be included in all code or typed directly into the SMI terminal before the motor can move.

```
EIGN(2)
EIGN(3)
ZS
```

After these commands are set, the LED light on the SmartMotor will change from solid red to flashing green indicating that the motor is ready and on standby to move.

4.3 Motion Commands

The Animatics SmartMotor can move using different modes including a torque mode, velocity mode, absolute position mode, and relative position mode. A basic absolute move example is shown below:

```
MP
VT=100000
ADT=1000
PT=20000
G
```

The velocity, acceleration, and position are given in encoder units. A conversion is therefore necessary to specify the position in inches, velocity in inches per second, and acceleration in inches per second squared. The Animatics SM23165DT used in this project rotates with 4000 encoder counts per revolution. Additionally the HLD60 Actuator with Internal Rollers has a 12.5 mm displacement per revolution.

Using simple programs, we can achieve linear and sinusoidal motion, as depicted by Figure 17:

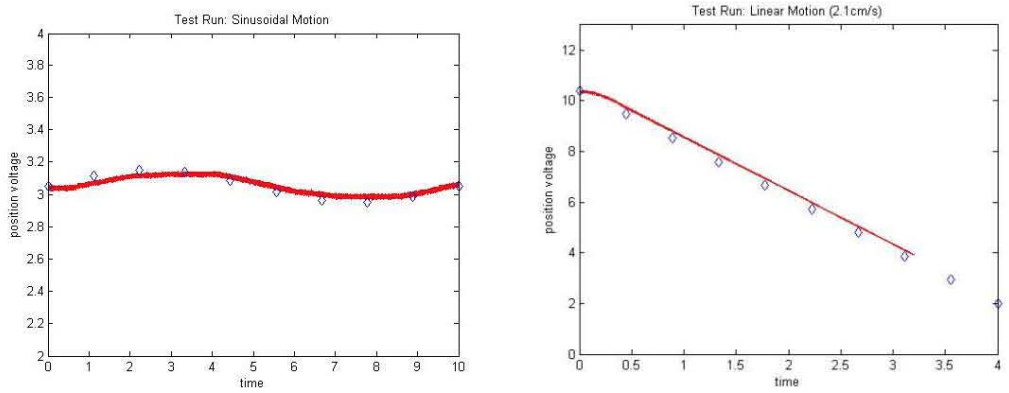


Figure 17: Linear and sinusoidal motion

However, SMI programs are only good for achieving relatively simple motion. A comprehensive list of desired position and/or velocity needs to be hardcoded into the program. This is not practical when we're ultimately trying to implement an arbitrary waveform. For example, if the waveform of velocity data is spaced at 20ms and has a duration of 30s, there are 1,500 velocity data points to write into the SMI program, which is not realistic or desirable.

4.4 I/O Ports

Upon further study of the User Manual, we discovered the SmartMotor's I/O Functions, which are extremely flexible and provide a variety of digital and analog input and output capability. Each I/O point has a corresponding pre-assigned variable name within the programming environment and can be read from, or written to, by placing it on the right or left side of an equation, respectively. The SmartMotor SM23165DT is equipped with 15 I/O points, which can be physically accessed by a DB-15 D-sub Connector. The pin numbers and their corresponding functions are listed in the following:

PIN	5V I/O Connector	Specifications:	Notes:	Diagram:
1	I/O – 0 GP or Enc. A or Step Input	25mAmp Sink/Source 10Bit 0-5VDC A/D	1.5MHz max as Enc or Step input	DB-15 D-sub Connector
2	I/O – 1 GP or Enc. B or Dir. Input	25mAmp Sink/Source 10Bit 0-5VDC A/D	1.5MHz max as Enc. Or Dir. Input	
3	I/O – 2 Positive Over Travel or GP	25mAmp Sink/Source 10Bit 0-5VDC A/D		
4	I/O – 3 Negative Over Travel or GP	25mAmp Sink/Source 10Bit 0-5VDC A/D		
5	I/O – 4 GP or RS-485 A Com(1)	25mAmp Sink/Source 10Bit 0-5VDC A/D	115.2KBaud Max	
6	I/O – 5 GP or RS-485 B Com(1)	25mAmp Sink/Source 10Bit 0-5VDC A/D	115.2KBaud Max	
7	I/O – 6 "G" command or GP	25mAmp Sink/Source 10Bit 0-5VDC A/D	Redundant connection on Main Pwr Connector	
8	Phase A Encoder Output	25mAmp Sink/Source		
9	Phase B Encoder Output	25mAmp Sink/Source		
10	RS-232 Transmit Com(0)		115.2KBaud Max	
11	RS-232 Receive Com(0)		115.2KBaud Max	
12	+5VDC Out	50mAmps Max		
13	Ground			
14	Ground			
15	Main Power: +12.5VDC to +48VDC	If -DE Option, Control Power separate from Main Power	With -DE option, this becomes separate control power input.	

Figure 18: SmartMotor Connector Pinouts

The 5V I/O is push-pull. To use the I/O functions, in our code we must first deactivate default on-board I/O functions. To take inputs, the following command is useful:

$$\mathrm{INA}(V1, \mathrm{exp}),$$

where "exp" is the I/O Bit Number or Status Word Number. This can be passed in from a variable.

4.5 Analog Velocity

From the User Manual, we found the following sample code, based upon which we were able to achieve motion based on an analog output of velocity:

```
EIGN(W,0)       'Disable hardware limits
KP=3020         'Increase stiffness from default
KD=10010        'Increase damping from default
F               'Activate new tuning parameters
ADT=100         'Set maximum acceleration
MV              'Set to Mode Velocity
d=10            'Analog dead band, 5000 = full scale
o=2500          'Offset to allow negative swings
m=40            'Multiplier for speed
w=10            'Time delay between reads
b=0             'Seed b
C10             'Label to create infinite loop
a=INA(V1,3)-o   'Take analog 5Volt FS reading
x=a-b           'Set x to determine change in input
IF x>d          'Check if change beyond deadband
  VT=b*m        'Multiplier for appropriate speed
  G             'Initiate new velocity
ELSEIF x<-d     'Check if change beyond deadband
  VT=b*m        'Multiplier for appropriate speed
  G             'Initiate new velocity
ENDIF           'End If statement
b=a             'Update b for prevention of hunting
WAIT=w          'Pause before next read
GOTO10          'Loop back to label
END             'Obligatory END (never reached)
```

A deadband is an interval of a signal domain or band where no action occurs (the system is dead). For our purpose, however, the input are waveforms that could be considered digital (signal is constantly changing) and thus setting the deadband to 10 is detrimental to obtaining consistency. We changed d to 0.

To achieve a maximum speed of around 0.1 m/s, we experimented with the multiplier value and changed it from 40 to 95. The wait time is decreased to 5 ($w = 5$) to accommodate higher sampling rates.

The most important line of this sample code is the following:

$$a = \mathrm{INV}(V1,3) - o$$

$\mathrm{INV}(V1,3)$ means we are retrieving input from I/O point 3, or pin number 4. Voltage is also scaled in millivolts where 3456 would be 3.456V. Setting the offset to 2500 ($o = 2500$) means the 2.5V is the zero-velocity voltage.

This code causes the SmartMotor's velocity to track an analog input.

5. Data Acquisition System

Data acquisition plays an important role in connecting Matlab and the SmartMotor. It queues the desired waveform sent from Matlab in an array format. While the SmartMotor is activated, its code retrieves the line of data to use as the velocity input. The basic setup is shown in Figure 19:

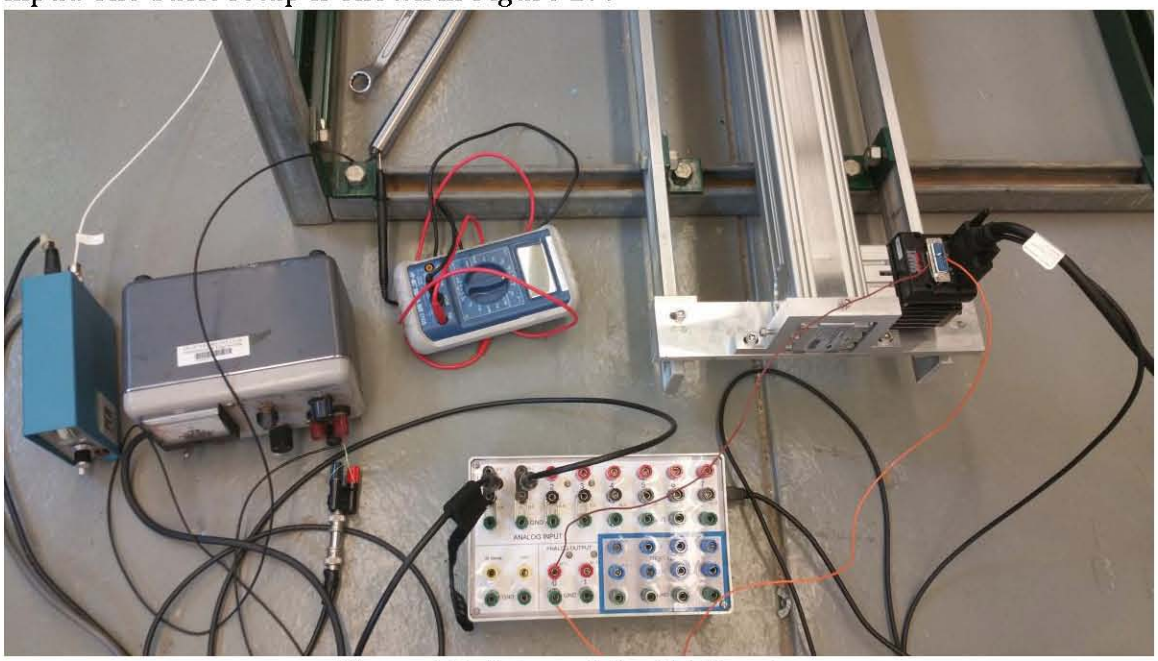


Figure 19: Setup of the DAQ system

As shown in Fig. , the analog output portion of the A/D converter is then connected to Pin 4 and 'ground' of the SmartMotor's I/O port. On the analog input section of the convertor, two connections go the accelerometer and LVDT's power source, respectively.

It is largely realized through Matlab's data acquisition toolbox. With the toolbox we can configure data acquisition hardware and read data into MATLAB and Simulink for immediate analysis. We can also send out data over analog and digital output channels provided by data acquisition hardware.

In Matlab, we create a session object that we can configure to perform operations using a CompactDAQ device:

$$w = \text{daq.createSesion}('ni'),$$

where 'ni' is the name of the vendor.

To collect data from our LVDT and accelerometer, we added two analog input channels to Matlab's DAQ session. To store data, we created one analog output

channel. After the desired waveform is generated inside a vertical array, we transfer it into the DAQ through the following command in Matlab:

$$\text{queueOutputData}(w, \text{motion})$$

To collect the data that the sensors collect during motion, we used the following command:

$$\text{data} = w.\,\text{startForeground}(\quad);$$

'Data' will consist of two columns: acceleration and displacement, both in terms of voltage. Using the calibration ratios given in the Introduction section, we can obtain the acceleration and speed in terms of SI units:

$$a = \text{data}(:,1) * 98.1 \ldots \ldots (m/s^2)$$
$$x = \text{data}(:,2) * .01 \ldots \ldots (m)$$

6. Calibration

While we transfer data from Matlab to the SmartMotor through the DAQ system, the array is seen as voltages, not velocity in SI units. Calibration is needed so we can input waveforms of velocity in terms of SI units (m/s).

We took a sample test run. Fig. . depicts time and position voltage on the two axes. By calculating the slope through extracting two data points, we found the actual velocity to be 2.1 cm/s. The voltage fed into the DAQ was 3.04V. Taking into account the 2.5V offset in our analog velocity code, we calculated the conversion ratio from SI units to SmartMotor voltages:

$$\text{calibration ratio} = \frac{.021m}{(3.045 - 2.5)V} = .0385$$

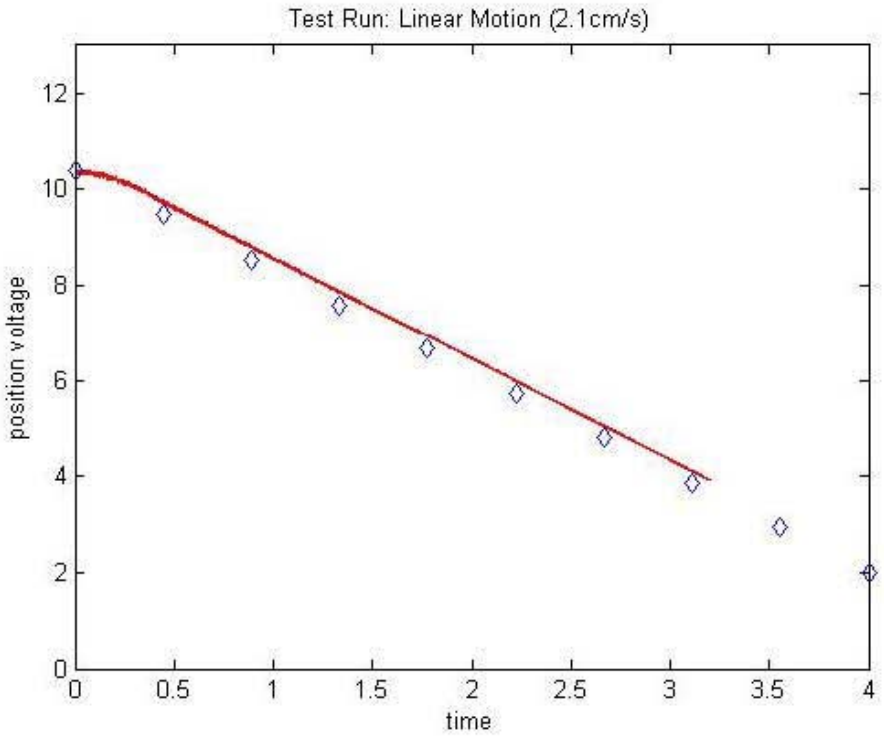


Figure 20: Test Run for calibration

Thus, in our Matlab code, we added the following line for unit conversion:

$$\text{motion} = 2.5 + \frac{[v]}{.0385},$$

where $[v]$ is the vertical array of analog velocity of the waveform to be queued into the DAQ system.

7. Testing Structures

7.1 Lumped Mass System:
The lumped mass system used is shown in Figure 21:

Figure 21: Lumped mass structure used for testing

The second moment of area is $I = \frac{1}{12}bh^3 = \frac{1}{6144}in^4$. The elastic modulus of aluminum is $E = 10(10^6)$psi, the weight of the aluminum block (with the attached SparkFun accelerometer and Arduino) is $W = 1.905$ lbs, the length of the cantilever beam is $L = 26''$. The spring constant of the lumped mass system is $k = \frac{3EI}{L^3} = .278\frac{\text{lbs}}{\text{in}}$. The resonant frequency is $\omega_0 = \sqrt{\frac{kg}{W}} = 1.19$ Hz.

7.2 Two-story Stick Mass Model

The model is shown in Figure 22:

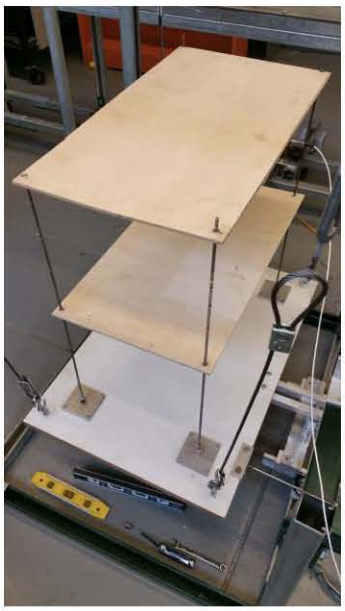


Figure 22: Two-story Stick Mass System

The columns are aluminum all-threads with a diameter of 0.186". The two stories are made from 1/8'' plywood, and weigh 1 lb. They are equally spaced along the column, with each story 8'' in height. At the support, the aluminum square plate had ¼'' tapped holes, which was bigger than the diameter of the selected all-thread. A special nut was made on the lathe to fit both the tapped hole on the aluminum square plate as well as the all-thread. 10-32 nuts were used to stabilize the two stories so they wouldn't move along the column, even during high frequency motion. Details of the special nut and the 10-32's are shown in Figure 23:

Figure 23: Connections details of the two-story stick mass system

As shown in the Theory section, the resonant frequencies are $w_1 = 8.69$Hz; $w_2 = 20.59$ Hz. The three-story stick mass system is identical in nature, and therefore is not shown in this report.

8. Result

8.1 The Big Picture

The system outline of the shake is shown below in Figure 24:

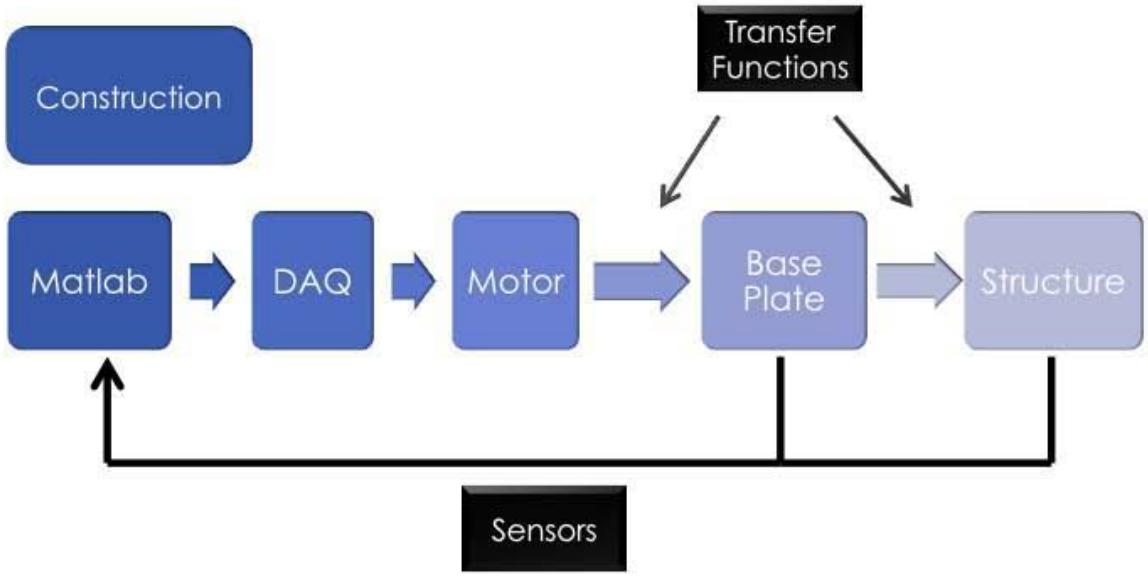


Figure 24: The Big Picture

8.2 Comparing command input with measure velocity output

After we calibrated the SmartMotor's analog velocity program, we experimented with a variety of input. We queued an impulse, sinusoid, and the El Centro into the A/D converter and recorded the base plate's position using the LVDT. We then plotted both the differentiated LVDT data (to obtain velocity) with the input velocity against time.

Comparing command input with measured velocity
For an impulse, the Matlab command is:

```
ones(10,1)*0.092;
```

Since the sample rate of the created DAQ session is 100 Hz, our input impulse has a duration of 10/100=0.1s at .092 m/s. At other times, the input velocity is zero. The described waveform is depicted precisely by blue in Figure :

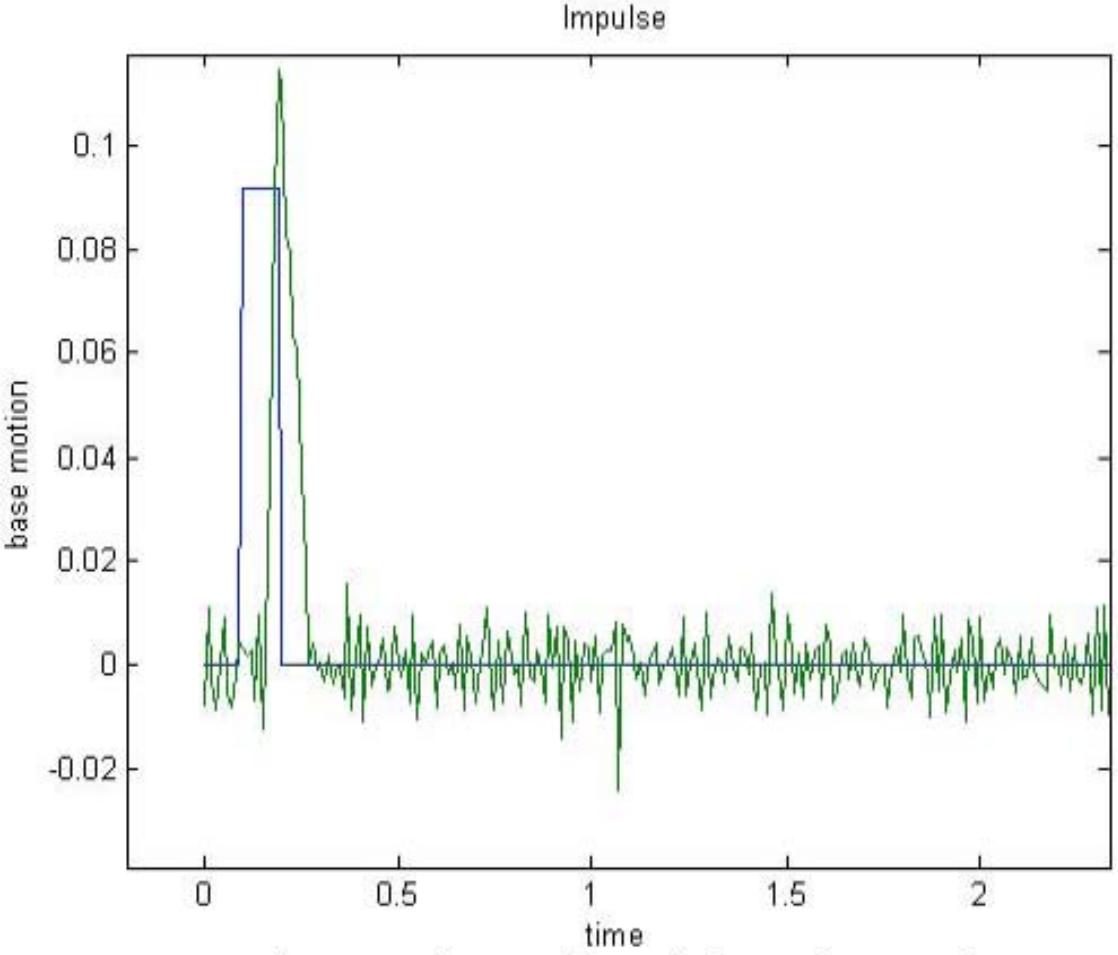


Figure 25: Command input and recorded input for an impulse

The more noisy green data is the position data measured by the LVDT, obtained from our DAQ's second analog input channel (the first input channel is reserved for the acceleration data, which is mentioned later). Note that there is an approximately 70 ms delay at the base plate's first sign of acceleration. At this point we can not evaluate the significance of this time lag without quantifying the transfer function between the motor and the base plate. We will now refer to that transfer function as H_{plate}.

For a sinusoid, the Matlab command is:

```
ssl = .062*sin(18.* time');
```

Like with the impulse, we plotted the input waveform along with the differentiated LVDT data to obtain Figure :

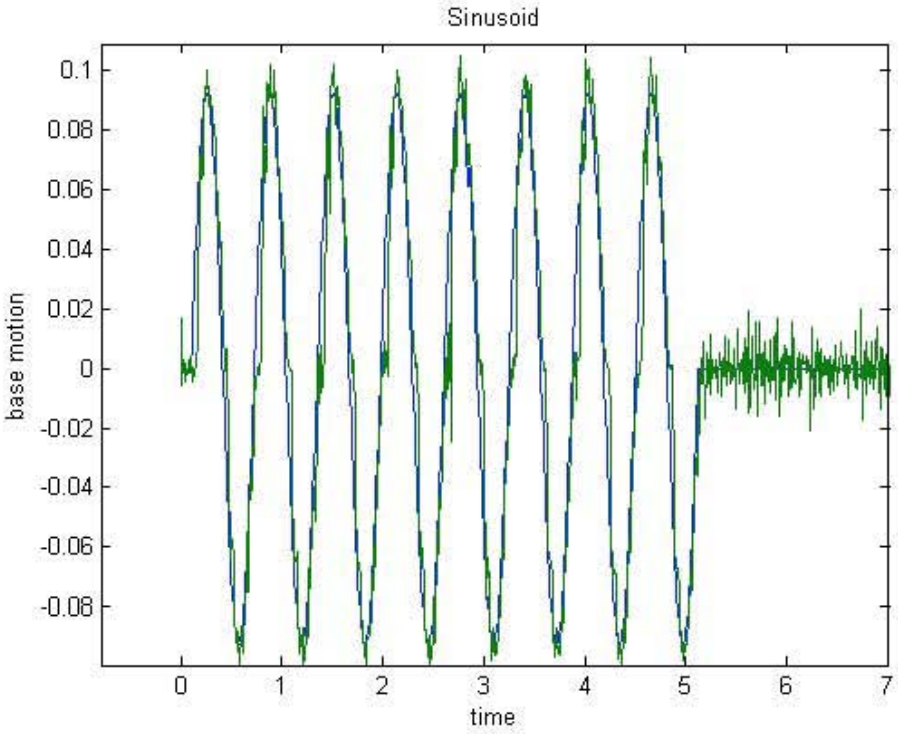


Figure 26: Command input and recorded input for a sinusoid

There is an apparent time lag, but otherwise the two match perfectly in terms of peak amplitude and period.

For the El Centro, we obtained the El Centro north-south acceleration data, and then performed Euler integration to get velocity (sample code attached in Appendix B). The Matlab command to create the input waveform is:

```
v_el(1:500);
```

We then plotted the input waveform along with the differentiated LVDT data to obtain Figure:

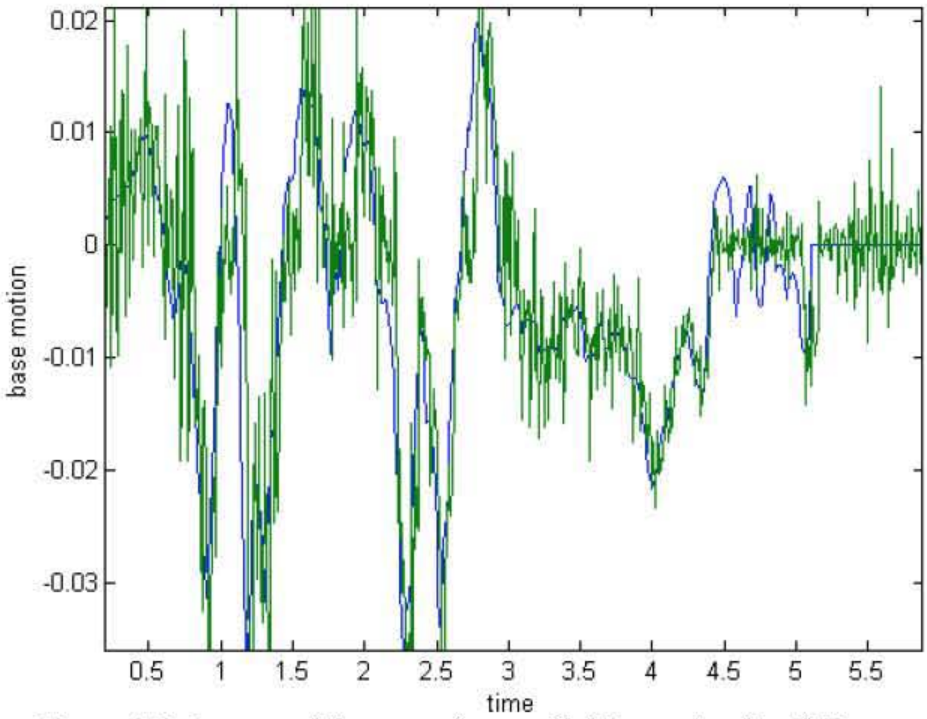


Figure 27: Command input and recorded input for the El Centro

8.3 Obtaining H_{plate}

To gain a close look, we zoom in on the first 100 ms of the impulse input/LVDT result (Figure):

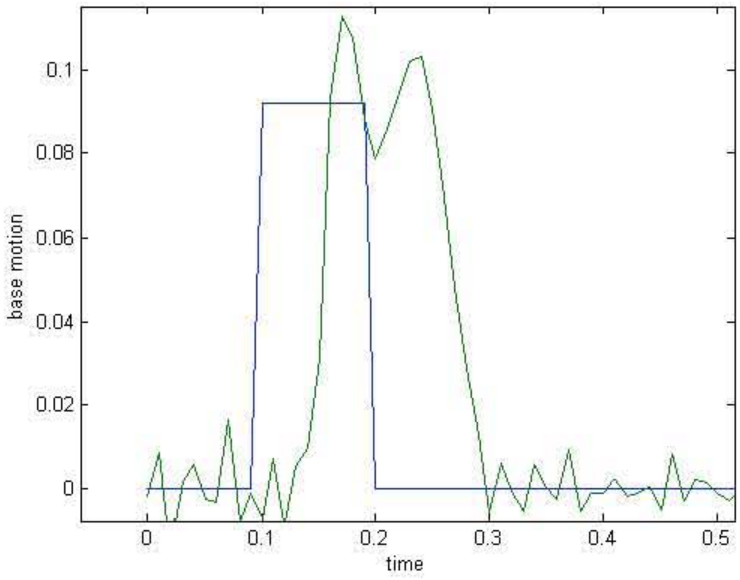


Figure 28: Zoom-in of Figure 27

Create arrays for the time and base plate position and use Matlab's curve fitting toolbox:

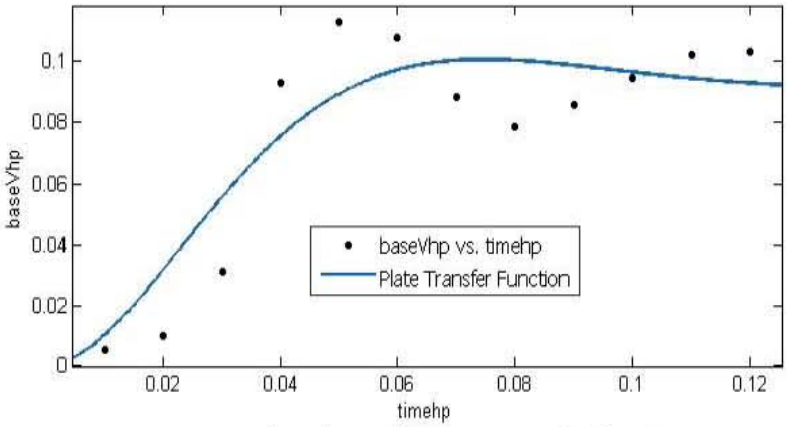


Figure 29: cftool to obtain curve fit for H_{plate}

The curve fitting results are as follows:

$$\omega_0 = 52.7; \zeta = 0.60; \tau = \frac{1}{\omega_0 \zeta} = 0.03 \ (R^2 = 0.81)$$

Note that H_{plate}'s time constant τ is only 0.03 seconds, which also translates to a cutoff frequency of around 5 Hz.

8. 4 Obtaining $H_{\text{structure}}$

To obtain the transfer function of the lumped mass structure, we applied the same impulse and obtained the following acceleration output results:

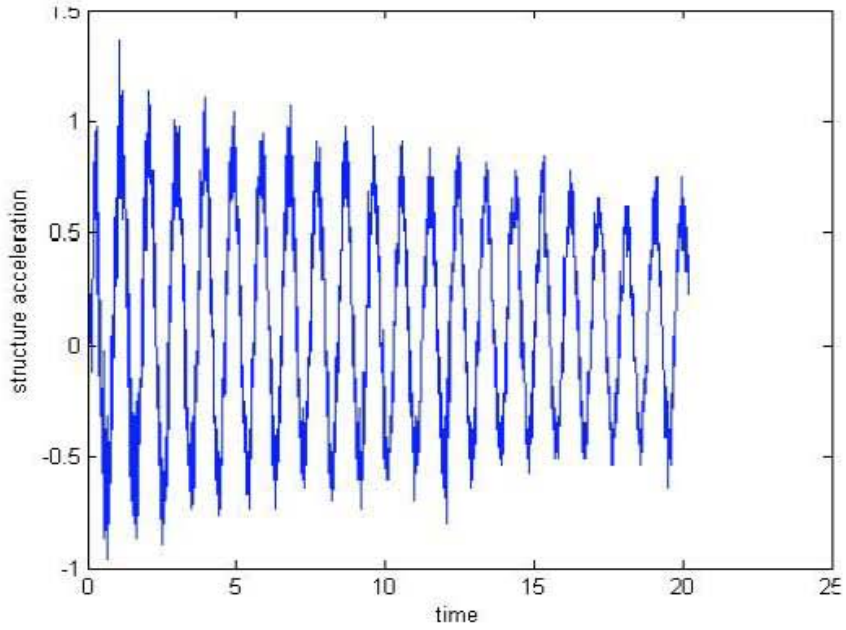


Figure 30: Response to Impulse

Create arrays for the time and structure acceleration and use Matlab's curve fitting toolbox:

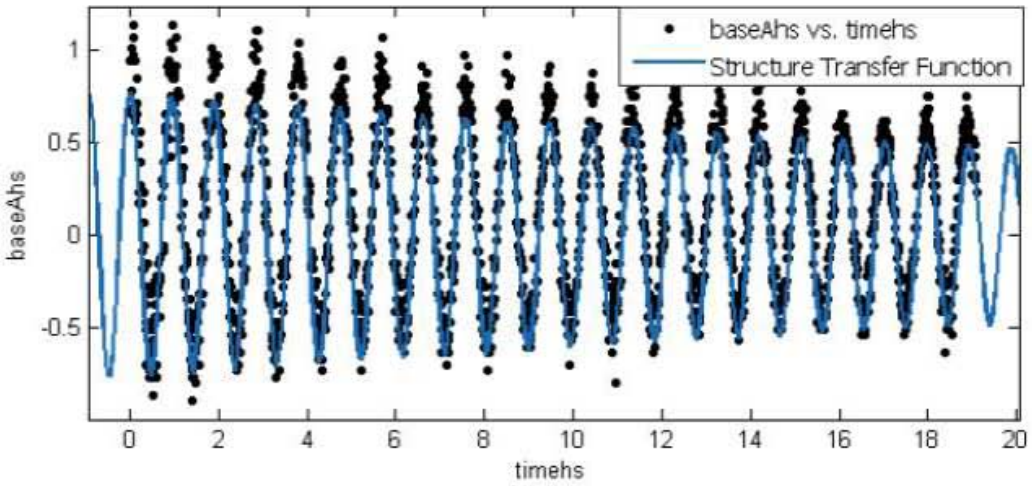


Figure 31: Curve fit for $H_{\text{structure}}$

The curve fitting results are as follows:

$$\omega_0 = 6.6; \zeta = 0.0036; \tau = \frac{1}{\omega_0 \zeta} = 42s \ (R^2 = 0.87).$$

Notice the time constant is much bigger than that of H_{plate}. This give us a valid reason to ignore H_{plate}.

8.5 Discussion on the significance of H_{plate}

The fact $\tau_{structure} \gg \tau_{plate}$ gives us one reason to ignore H_{plate}. Also recall that its time constant corresponds to a cutoff frequency of 5 Hz. Plotting a vertical line of 5 Hz on the frequency-amplitude of El Centro:

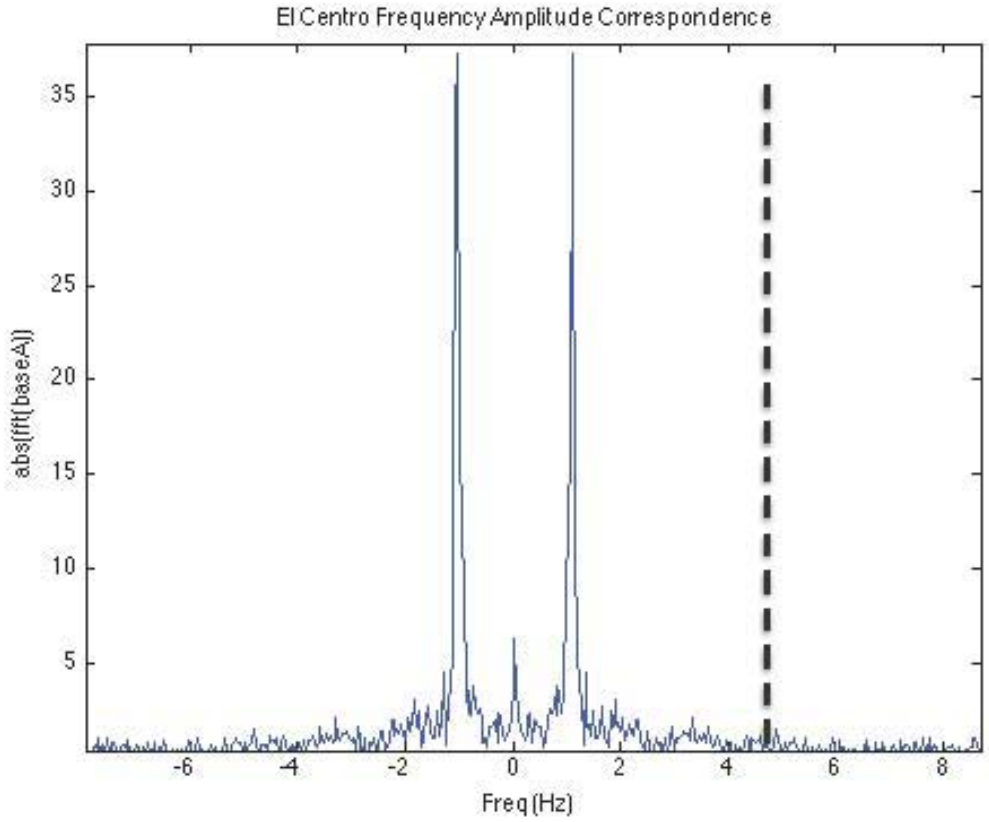


Figure 32: Content of El Centro captured by H_{plate}

Observe the majority of the earthquake's amplitude are in the low frequency region, and are captured by the base plate's transfer function. This points to another valid reason to ignore it.

Based on the fact that the base plate's transfer function has a significantly smaller time constant compared to that of the structure's, and that its cutoff frequency is higher enough to capture the majority content of the El Centro (in fact most major earthquake behave similar to El Centro, in that the majority of their content lies in the low frequency region), we have decided to ignore the base plate's transfer function, H_{plate}.

8. 6 Using Matlab's LSIM

`lsim` simulates the (time) response of continuous or discrete linear systems to arbitrary inputs.

`lsim(sys,u,t)` produces a plot of the time response of the dynamic system model sys to the input history, `t,u`. The vector `t` specifies the time samples for the, and consists of regularly spaced time samples. The input `u` is an array having as many rows as time samples and as many columns as system inputs. For instance, if `sys` is a SISO system, then `u` is a t-by-1 vector. If `sys` has three inputs, then `u` is a t-by-3 array. The model sys can be continuous or discrete, SISO or MIMO.

Since our system is a discrete SISO, both `u` and `t` have only one column. After obtaining the structure's transfer function, $H_{structure}$, we obtained the theoretical acceleration output for three types of command input (impulse, sinusoid, and El Centro) to see how well they compare.

For the Impulse input, the `lsim` result is in blue, while the measured acceleration by the accelerometer is in green:

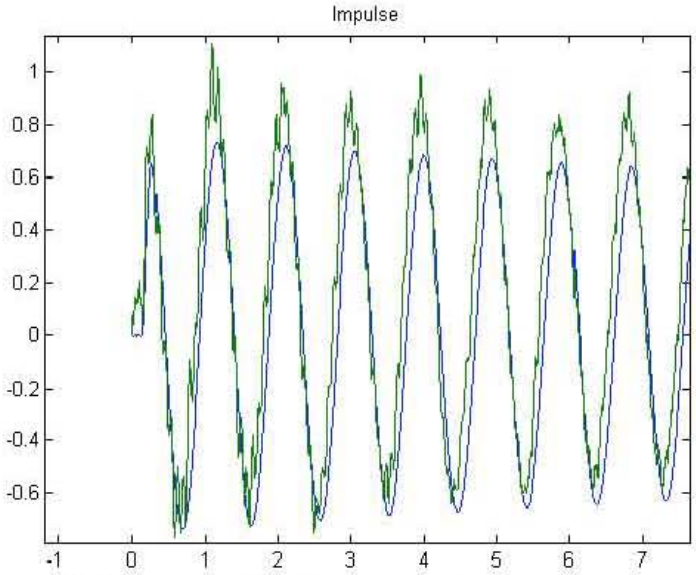


Figure 33: Theoretical and measured output for an impulse

For the Sinusoidal input, the `lsim` result is in blue, while the measured acceleration by the accelerometer is in green:

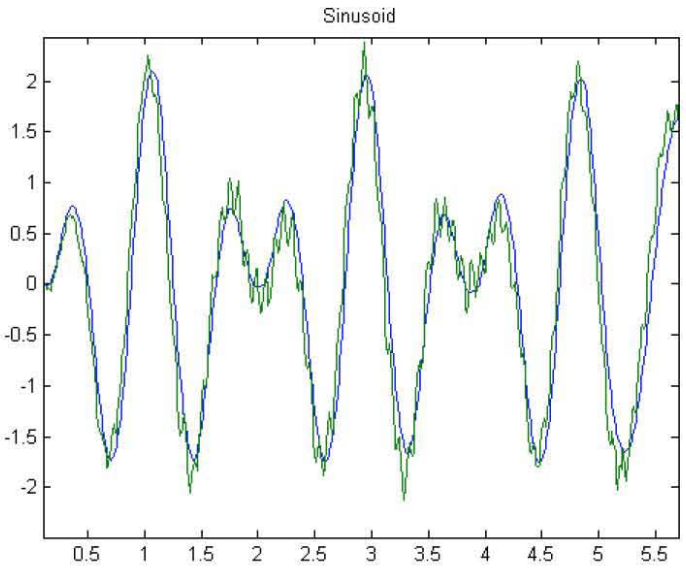


Figure 34: Theoretical and measured output for a sinusoid

For the El Centro, the `lsim` result is in blue, while the measured acceleration by the accelerometer is in green:

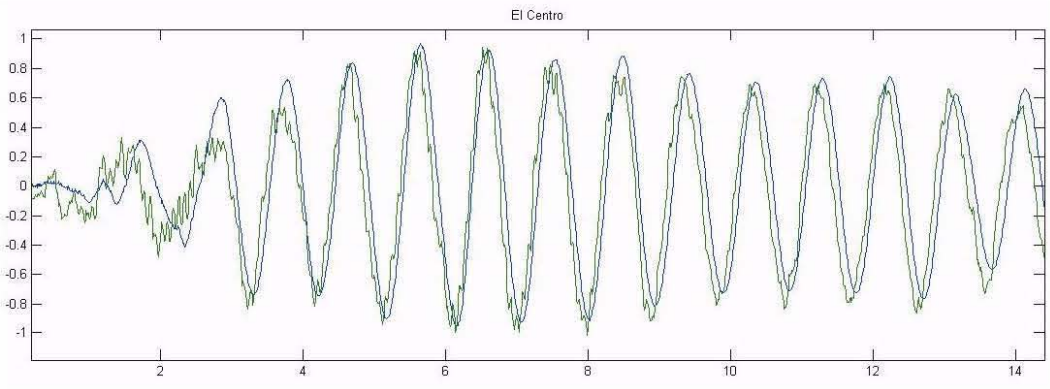


Figure 35: Theoretical and measured output for El Centro

We see that the `lsim` results for impulse and sinusoid input are good in in terms of amplitude and frequency. They are both for relatively short durations. For the El Centro output, we can see the basic shape of the predicted result matches that of the measured acceleration. However, there is a noticeable shift throughout the entire waveform starting before the first second.

8. 7 Addition of a dashpot to enhance the El Centro simulation results

We then speculated the cause of the shift for the `lsim` results for the El Centro. Recall that our structure system is very lightly damped, with a damping coefficient of near zero ($\zeta = .0036$) and the time constant of over 30 seconds. Such light damping could cause the delay in response, which is then amplified by the long duration of the El Centro waveform.

To verify our speculation, we added a dashpot stand onto our frame (again, made easy by choice of our framing material—Unistrut™). The rod of the dashpot is fastened onto the structure by gluing a No. 5 nut onto its front surface. There is a even great limit to the amount of possible displacement as the dashpot's rod is even shorter than that of the LVDT. Due to this limitation, we could only run a five second waveform of the El Centro.

First, we found the transfer function for our more heavily damped system by applying an impulse and measuring the acceleration response using the accelerometer:

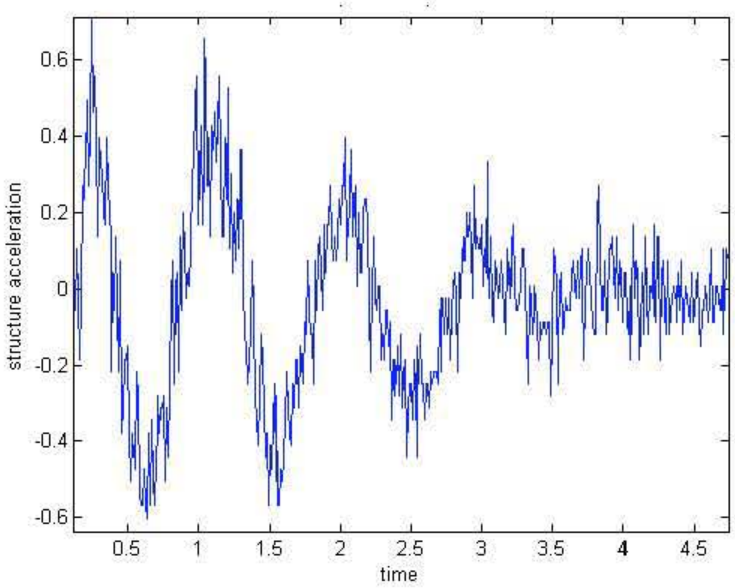


Figure 36: Response to an impulse for more heavily damped system

We then imported this data into Matlab's curve fitting toolbox and found the following results:

$$\omega_0 = 7.267; \zeta = 0.08; \tau = \frac{1}{\omega_0 \zeta} = 1.7 \ (R^2 = 0.38)$$

Note the increased damping coefficient and the significantly smaller time constant. We then coded this transfer function into Matlab's workspace and performed `lsim` on just five seconds of El Centro:

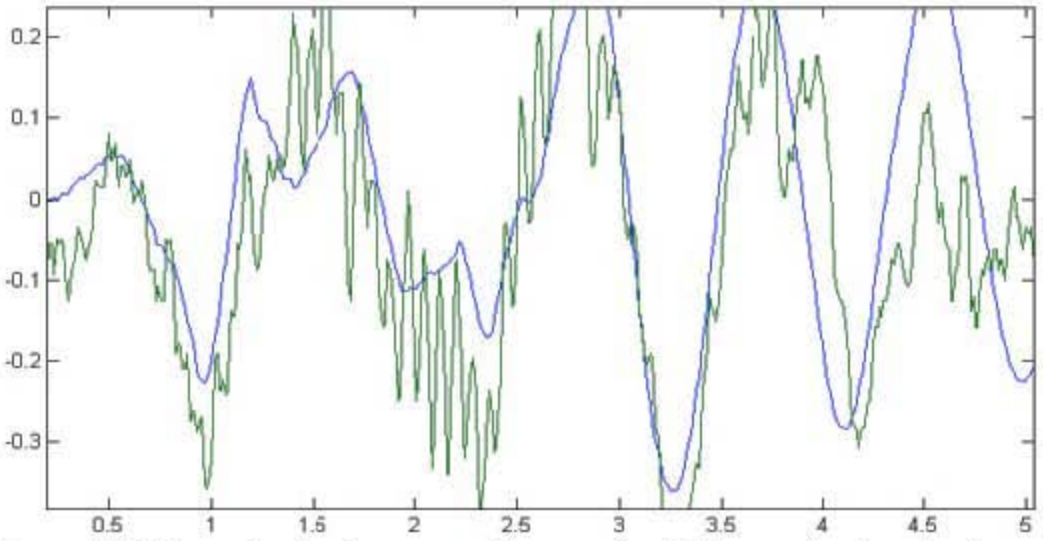


Figure 37: Theoretical and measured output for El Centro for heavily damped system

Note that shifting only started after the third second mark, which shows an improvement from the previous system.

8.8 FFT analysis of two multistoried stick mass structure

We constructed a twos-story stick mass model, as shown in the Theory section. Each floor was sawed from 1/8" plywood at the woodshop, then drilled with a No. 12 drill at the four corners, which are spaced to match the four aluminum square plates on the four corners of the shake table. 32-thread/inch aluminum all-threads with a .186" diameter was used as the columns. Connection details are also shown in the Theory section.

To find the structure's resonant frequencies, we shook it using an impulse. Our PCB accelerometer was fixed onto a nut glued on the top floor. The acceleration response was plotted, and then we applied the FFT procedure to obtain the frequency domain information:

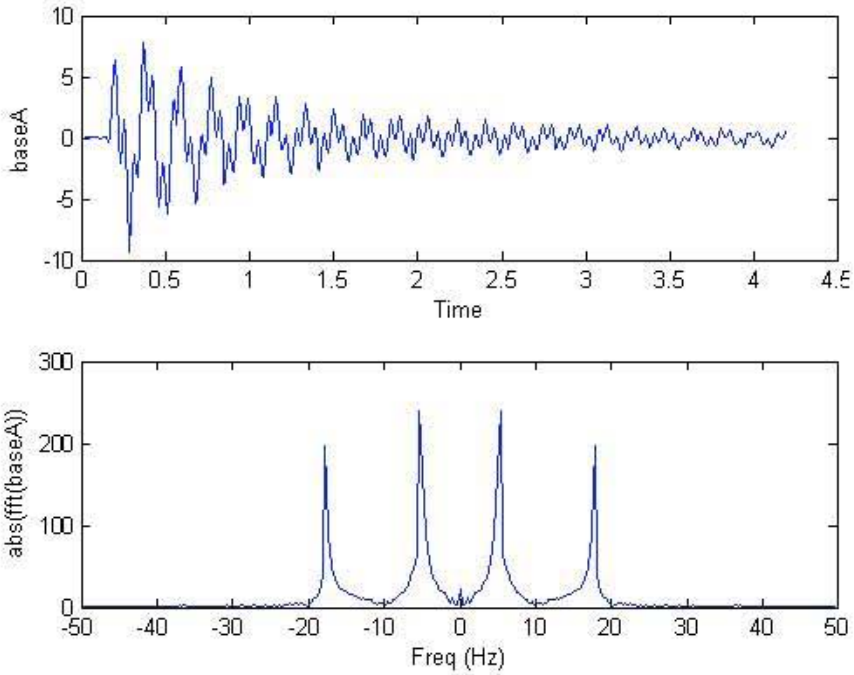


Figure 38: Time and frequency domain responses to an impulse of a 2-story structure

The two resonant frequencies are 6.1 Hz and 19.2 Hz. This is a standard procedure for finding the resonant frequencies of the multi-story structures. We applied the same approach to a three-story stick mass structure:

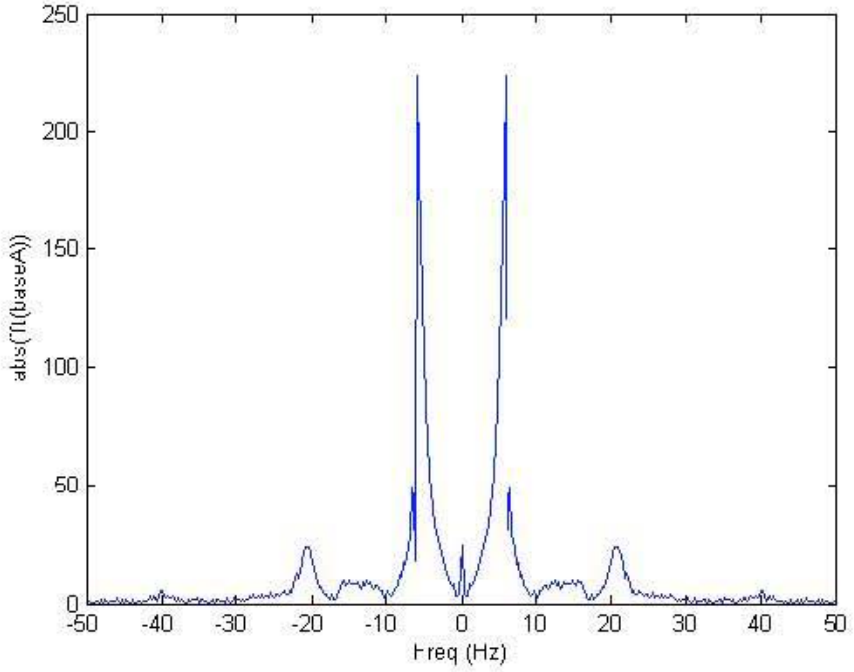


Figure 39: Time and frequency domain responses to an impulse of a 3-story structure

9. References

Engineering 12 Website, Erik Cheever

Structural Dynamics—Theory and Computation, Mario Paz

MOOG Animatics. (2014). SmartMotor User's Guide: Class 5 SmartMotor Technology. Moog Animatics.

Unistrut. (2013). Unistrut. Retrieved January 2014, from Atkore International, Inc.: http://www.unistrut.us/

Appendices

Appendix A
Analog Output

```
% Analog velocity output to simulate waveforms
% Bill Wu ENGR 90

beep
disp('xxx');
specify_position;
pause(3);
beep
pause(0.1)
disp('yyy');

clf
clear
clear global

w = daq.createSession('ni');
w.Rate = 100;
% No duration in seconds required with analog output

addAnalogOutputChannel(w,'Dev3','ao0','Voltage');
addAnalogInputChannel(w,'Dev3', 'ai0', 'Voltage');
addAnalogInputChannel(w,'Dev3', 'ai1', 'Voltage');

w.Channels(1).Range = [-10 10];
w.Channels(2).Range = [-10 10];

load('elcentrino.mat')
a_el=-a;
dt=t(2)-t(1);
v_el=cumsum(a_el-mean(a_el))*dt;
x_el=cumsum(v_el-mean(v_el))*dt;

% Impulse
% s = ones(10,1)*0;
% ss=ones(10,1)*0.092;
% r = ones(400,1)*0;
% v = [s;22;r];

% Sinusoid
time = 0:dt:10;
ss1 = .04*sin(1 .* time');
ss2 = .04*sin(2 .* time');
```

```
ss3 = .04*sin(3 .* time');
ss4 = .04*sin(5 .* time');
ss5 = .04*sin(7 .* time');
v = [s;ss1;ss2;ss3;ss4;ss5;r];


%     cf:     0.04*(1-1/sqrt(1-z^2)*exp(-z*w*x)*sin(w*sqrt(1-
z^2)*x))
%   cff:   c*w^2*exp(-x*w*z)*(cosh(x*w*(z^2   -   1)^(1/2))   -
(z*sinh(x*w*(z^2 - 1)^(1/2)))/(z^2 - 1)^(1/2))

% El Centro
% v = v el(1:1000);

Basemotion = 2.47 + v/0.0385;

queueOutputData(w,Basemotion);

s = serial('COM3'); % creates serial port object for motor
% set all the properties of the port
s.BaudRate           = 9600;
s.DataBits           = 8;
s.Terminator         = 'LF/CR';
s.RequestToSend      = 'off';
s.FlowControl        = 'software';
s.DataTerminalReady  = 'off';
s.parity             = 'none';

fopen(s); % connect serial port object to the motor
disp('Connected to motor')

% remove hardware limits, set velocity and acceleration
initialize = ['EIGN(2) EIGN(3) ZS'];
fprintf(s, initialize)

go = ['RUN'];
fprintf(s, go)

%Use  the  startForeground  function  to  start  the  analog
output...
%operation  and  block  MATLAB  execution  until  all  data  is
generated.

data = w.startForeground();

% After excution of analog output, tell motor to rest
command = ['MV VT=0 ADT=200 G END'];
```

```
fprintf(s, command)

fclose(s); % close serial port cleanly

hold off
pause(1)
time = ((1:size(data,1))-1)/w.Rate;
baseA = data(:,1)*98.1;
baseP = data(:,2)*0.01;
baseV=diff(baseP)./(diff(time'));
baseV(end+1)=baseV(end);
%baseV = smooth(baseV);

plot(time,v,time,baseV),xlabel('time'),ylabel('base
motion')
%legend('Command','Output');

%subplot(2,1,1),
plot(time,v),xlabel('time'),ylabel('Command')
%subplot(2,1,2),
plot(time,baseA),xlabel('time'),ylabel('Output')
figure
plot(time, baseA)
% figure
%          plot(time,data(:,1)),xlabel('time'),ylabel('accel
voltage')
beep
disp('zzz')
```

Appendix B
FFT code

```
load two story
subplot(2,1,1);
plot(time,baseA);
xlabel('Time'); ylabel('baseA');

subplot(2,1,2);
bFFT = fftshift(fft(baseA));
dt=time(2)-time(1);
f=((1:length(time))-(length(baseA)/2))/dt/length((baseA);
plot(f,abs(bFFT))
xlabel('Freq (Hz)'); ylabel('abs(fft(baseA))');
```